Combining Reinforcement Learning with Supervised Deep Learning for Neural Active Scene Understanding

Dano Roost[1], Ralph Meier[1], Giovanni Toffetti Carughi[1], and Thilo Stadelmann[1,2]

***Abstract*— While vision in living beings is an active process where image acquisition and classification are intertwined to gradually refine perception, much of today's computer vision is build on the inferior paradigm of episodic classification of i.i.d. samples. We aim at improved scene understanding for robots by taking the sequential nature of seeing over time into account. We present a supervised multi-task approach to answer questions about different aspects of a scene such as the relationship between objects, their quantity or the their relative positions to the camera. For each question, we train a different output head which operates on input from one shared recurrent convolutional neural network that accumulates information over time steps. In parallel, we train an additional output head using reinforcement learning (RL) that uses the reduction in cumulative loss from the supervised heads as reward signal. It thereby learns to gradually improve the prediction confidence of e.g. partially occluded objects by moving the camera to a more favourable angle with respect to these objects. We present preliminary results on simulated RGB-D image sequences that show superior performance of our RL-based approach in answering questions quicker and more accurately than using static or random camera movement.**

I. Introduction

The computer vision revolution of recent years has been driven in great parts by convolutional neural network (CNN)-based object detection approaches that can find and label entities in images [33], [17], [40]. However, the capabilities of these systems are still far from actual human perception. This is for example evident in (a) the absence of specific memory in wide-spread models such as [37], [18] that would allow a system to accumulate knowledge over multiple time steps and lead to object permanence [28]; or (b) the inability to actively move the camera in order to improve the current understanding of the scene and increase the chances for a successful classification of present objects [16]. Actively controlling the camera to obtain a deeper understanding of the environment would be especially valuable in modern robotic use cases such as [7], [3], where perception and action are closely connected and an advanced understanding is necessary that goes beyond pure detection and classification. While traditionally handcrafted mappings or SLAM techniques have been used [11], the rise of deep neural networks enables new possibilities that allow for going more human-like paths [27], [5], [13]. This stimulates a fundamental rethinking of the predominant approach to computer vision that works on i.i.d. samples [16] by rather taking inspiration from the human brain [34].

Research suggests that the internal scene representation of the human brain is created by different recurrent connections between the primary visual cortex (V1) and regions like the lateral geniculate nucleus (LGN) or the lateral occipital complex (LOC) [36], [10]. This representation forms the basis for two different streams of processing, that, roughly speaking, perform object classification on the one hand (the "ventral stream" that is hierarchically organized to classify a large number of shapes using different levels of abstraction) and object localization on the other hand ("dorsal stream" with limited temporal memory that is not needed for short-term spatial mapping) [35].

In this paper, we present an active vision approach based on a neural network architecture that is inspired by the above-mentioned properties of the human visual system: a backbone consisting of a CNN-recurrent neural network (RNN) enables hierarchical scene decomposition and attention over multiple time steps; it feeds its learned representation into multiple output heads to perform different tasks like localizing and classifying objects as well as to control the movement of the camera for the next time step. Our main contributions are (a) a novel multi-headed deep neural architecture to achieve object permanence for improved scene understanding; and (b) a combined supervised and reinforcement learning (RL) approach to end-to-end train the system by feeding the overall reduction of loss of the supervised heads as the reward signal into the RL-based camera control head. Our results on rendered scenes of primitive objects show significant improvement over passive vision systems when answering questions like "what object is stacked onto the red cube": higher success rate in answering correctly is reached with much less subsequent frames, i.e., quicker and better perception is performed.

II. Related work

A. Neural Scene Understanding

Traditionally, the field of scene understanding is concerned with acquiring a representation of a scene that makes downstream specialized tasks easier than if performed on raw sensory input [32]. Creating such an intermediate representation using neural networks has been an active area of research in recent years. Eslami et al. [12] describe a system that creates such a representation using a type of variational autoencoder architecture. A shortcoming of this approach, however, is that it needs fully specified 3D models for all present objects. This is solved in their follow-up work [13]: with only a few

[1]The authors are with the ZHAW School of Engineering, Zurich University of Applied Sciences, Obere Kirchgasse 2, 8400 Winterthur, Switzerland: {dano.roost,ralphlmeier}@gmail.com {toff,stdm}@zhaw.ch

[2]Thilo Stadelmann is a Fellow of the ECLT European Centre for Living Technology, Venice, Italy

2D input images and their corresponding viewpoints in a 3D scene, the presented Generative Query Network (GQN) is capable of synthesizing renderings from previously unseen viewpoints within the scene. With an increasing number of input images, the quality of the synthesized output image improves and its uncertainty decreases. The approach uses two networks, the first one aggregating the data of the input images into a compressed scene representation and the second one producing a rendering based on this representation and an additional input, which is the desired camera position for rendering by sampling from the scene representation vector. The authors also use the learned representation as input for RL-based robotic grasping, achieving a four times higher sample efficiency than when using raw pixel input.

The resulting renderings, however, come at the price of requiring large quantities of training data. For each of the four presented experiments, at least 2 million training scenes have been used. Collecting comparable numbers of real-world data would hardly be achievable in reasonable time. Additionally, the process is simplified considerably by providing the system with coordinates of the collected input images. We think that a system should be capable of inferring this change in perspective on its own by using just temporally correlated images that have been acquired in sequence.

B. Active Vision

While a large number of use cases in computer vision assume a stationary camera or given image sequences, including [12], [13], the field of *active* vision deals with cases where the system is embodied in an active agent and can manipulate the viewpoint in order to perceive more complete information from the scene [1], [9], [8]. For humans, it is natural to catch glances from as many different angles as necessary, subject to availability, in order to reduce remaining uncertainty and thereby obtain a more complete mental representation of a scene or object. The same is useful for robots [26], [22].

Cheng et al. [4] for example apply RL for the combined control of a robotic gripper and an active camera. By using actor-critic algorithms [23], [29], their system can pick up a target object even in the presence of distractor objects that may occlude the direct line of sight. By training to move the camera and pick up the target, the system implicitly learns to understand how relevant a certain feature in the scene is. This process of focusing on important things while discarding less relevant input information is also known as visual attention [21], [42]. We think that visual attention and active vision are closely related because attention can steer the focus to areas which are not yet known well. Humans for example deal with areas of missing information by using eye movement and change of viewpoint as forms of active vision.

By focusing on specific areas of a scene, the process of active vision is very close to the attention mechanism described by Mnih et al. [30]. Here, the authors perform digit classification on the MNIST database [25]. Using a RNN, the focus is moved to specific sections of the input image in form of windows of different size all centered around a specific location: the inner-most window (having the smallest size) can be considered as the equivalent to the human foveal image. The outer windows get down-sampled to the same size as the inner-most one and, having then lower resolution, can be compared to peripheral vision. To avoid the need for a sliding window that scans the *whole* picture, the approach uses RL to find the next region to focus on. Our active vision approach builds on similar concepts, but instead of moving a window of attention around a pre-acquired image, it controls the actual acquisition in a feedback loop with object detection.

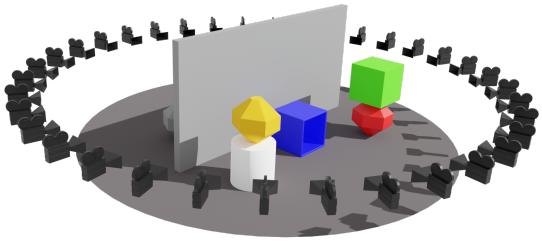

Fig. 1. An example scene with occlusion from our experiments, including the designated camera positions from which the scene can be observed.

III. Our Approach

A. Scene Setup

The proposed approach operates on sequences of RGB-Depth (RGB-D)-input images, each having a resolution of $4 \times 128 \times 128$ pixels. We create a synthetic test bed of training and test images using the 3D rendering software Blender [6] as follows: per scene, we perform a circular camera trajectory around a randomly arranged group of primitive objects and capture 36 images, all with the camera pointed towards the center of the group as visible in Figure 1. To simplify learning, we assign a unique combination of color and shape to each object, so that it can be clearly identified during the training period. We place some objects on top of each other (but not more than two). Notably, we do not provide any information regarding the camera position to the system. Instead, we expect the system to infer its own position based on the percept history in the current scene.

As training data, we create $20,000$ example scenes for each of two different basic scene setups: one with an occlusion object in the center of the scene (see Figure 1), and one without. Ground truth is obtained from the rendering software first as a global position for each object. Using the per-frame world matrix of the camera base, these positions can be transformed into relative positions. Besides this world matrix of the camera base, other information like the screen space position of each object is also saved for each frame. This allows us to query the same input sequence on a large number of different tasks such as an object's location ("what is on top of the blue cylinder?"), properties ("what shape does the object below the yellow cube have?") or number of occurrences ("how many objects are present in

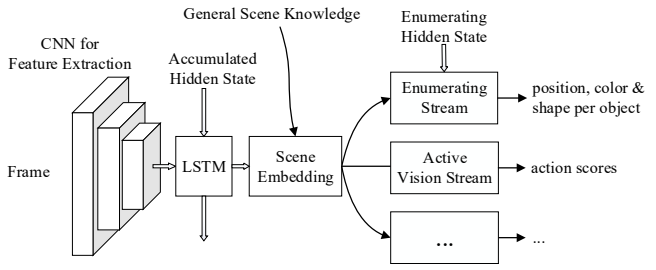

Fig. 2. Our network architecture accumulates information over time in the LSTM after extracting features using the CNN. The resulting scene embedding vector feeds different output heads (2 out of 7 streams shown).

the scene?"). This forces the system to acquire a higher order scene understanding instead of performing simple input-to-output mapping. Notably, the whole data generation process is performed prior to training purely for speed reasons. As this means that all possible viewpoints are pre-rendered, performing active vision in our setup means to chose a camera trajectory by sequentially selecting which image from the 36 available ones of the current scene to see next.

B. Accumulating Information About a Scene

To create a persistent scene representation over multiple time steps, we use a 5-layer CNN with LeakyReLu activation [41] for feature extraction and one layer with 2048 LSTM [19] units for accumulating information. This is followed by a block of 4 fully connected layers with LeakyReLu activation and batchnorm [20] before and after the block, which produces the 2048-dimensional scene embedding vector s. This vector is recalculated on every time step i and should contain more and more detailed information regarding the scene at hand. Scene embedding vector s_i is then picked up by the different task-specific output heads that we call "streams" (see examples in Figure 2) to answer questions about the scene like "given color and shape, at what position (relative to the camera base) can a respective object be found?". We overall train 7 different streams that are capable of answering questions about objects and their relations.

The most complex output head called the *Enumerating Stream* outputs all found objects with their positions, shapes and colors. As the number of objects in the scene is not always the same, we use another LSTM network within this head that outputs one object at a time and whether there are any more objects to unroll on each step. To pair ground truth and predicted objects we use the Hungarian method [24] to minimize the total distance between actual and predicted positions of all objects.

C. Active Vision as a Reinforcement Learning Problem

With the goal, environment and basic model architecture for scene understanding defined as above, we can now formulate the task of active vision as the RL problem of reducing the remaining amount of uncertainty in the system's answers with each time step as much as possible. We can define that uncertainty δ_i at time step i as the sum of the loss of all output heads at said time step:

$$\delta_i = \sum_{n=0}^{S} L_n^i \quad \text{with} \quad \begin{array}{l} S = \text{Number of streams} \\ L_n^i = \text{Loss of stream } n \text{ at time step } i \\ \delta_i = \text{Uncertainty on time step } i \end{array}$$

We use δ_i to encode the desirability of uncertainty reduction in the reward $\mathcal{R}_i$ at time step i as follows:

$$\mathcal{R}_i = \delta_{i-1} - \delta_i$$

During training, we now select a random starting point for every trajectory (scene) and give the RL algorithm 7 different actions U to choose from, where positive values indicate steps to the right and negative values steps to the left.

$$U = \{-5, -2, -1, 0 \text{ (stay here)}, +1, +2, +5\}$$

When for example the agent chooses the $+5$ action, the next input frame will be the one 5 steps to the right, relative from the current input image's camera position (we include the "stay here" option to give the system the opportunity to focus on different aspects of the same input). Then, we let the Q-learning algorithm [31], [39] operate on the scene embedding vectors s_i as representations of the current state.

To implement this, we add another output head with output size 7 (our action space) for the q-values $q(u, s)$ of action u in state s. At each step of a training episode, we take actions ε-greedy according to the current q-function with a decaying $\varepsilon_j = 0.9999 \cdot \varepsilon_{j-1}$ and $\varepsilon_0 = 1$ to trade off exploration with exploitation over the training epochs j. The q-function is updated after every episode based on the reward values from these Monte Carlo roll-outs up to the terminal time step T as follows:

$$q(u_i, s_i) \approx \sum_{t=0}^{T-i} \gamma^t \mathcal{R}_{i+t}$$

To update the network, we use the squared difference between the predicted q-value and the Monte Carlo roll-out reward as loss value.

The training process takes place simultaneously for the supervised and the RL stream(s) by running each episode for exactly 14 time steps. The calculated loss of the supervised streams is not only used for *their* training using backpropagation, but also serves as input for the reward calculation to improve the RL stream. With ongoing training of the supervised streams, this loss changes, which means that the RL algorithm needs to deal with a highly non-stationary environment. However, this should not be a problem for the algorithm as it does not rely on experience replay [15]. On the downside, this leads to a lower sample efficiency of the training since the possibility of reusing previously seen episodes is missing. Furthermore, it complicates the evaluation of the active vision results in isolation since improvements in the supervised tasks could be either due to improvements of the RL policy or to the ongoing training of the supervised streams themselves. Consequently, it is necessary compare any learned policy with fixed or static types of camera control to measure success.

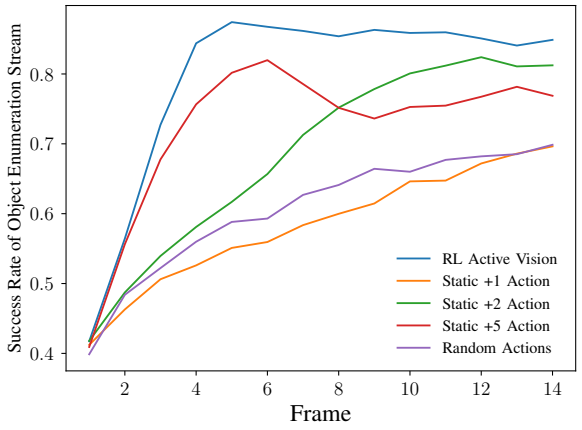

Fig. 3. Performance for different types of camera control. The proposed method outperforms all other types significantly, as seen in a higher success rate for object enumeration after the least amount of seen frames.

IV. Results

A. Active Versus Passive Vision

To evaluate the approach, we follow 14 steps-long trajectories over the 36 captured pictures for each scene using different methods for selecting the next frame. At each point in a trajectory, we evaluate the metric S we call "Success Rate" based on our most sophisticated output head, the Enumeration Stream:

$$S = \frac{\text{\#correctly identified}}{\text{\#correctly identified + \#incorrectly identified}}$$

Outputted objects that do not exist as well as objects that do exist but have not been found by the system count as incorrectly identified. Hence, the Success Rate is high when both the recall and the precision of the result are high. To measure the impact of the RL-based active vision, we perform an ablation study using the following alternative policies for camera control:

- *Random policy:* Randomly sampled actions from U.
- *Static policies:* On every time step, the camera moves a fixed number of steps to the right.
- *RL-based policy:* Greedy according to highest Q-value.

Figure 3 shows that the system is improved by using active vision, especially while seeing the first few frames. While the static policies with $+2$ and $+5$ also perform decently, the RL-based active vision achieves a head start and reaches the maximum success rate after having seen only 5 frames. Interestingly, the performance of the static $+5$-policy drops after the 6^{th} frame. This could be because the $+5$-policy actually performs more than one circle around the scene, potentially confusing the system. The most constant performance gain is achieved by the $+2$-policy. The proposed active vision approach however is outperforming all of them without relying on a fixed action for each time step.

B. Discussion

From this result, it can be concluded that the system is indeed implicitly capable of dynamically calculating the transformation matrix [14] to situate newcoming images into the existing scene embedding, independent of the amount of change in view point, and does not rely on any statically learned matrix. Further research would however be required to evaluate whether a similar performance would be possible by following a trajectory that does not have the circular shape as in the presented experiment. With data from this experiment alone, however, it is evident that given enough time and relatively simple camera trajectories, our system can accumulate most of the desired information regarding object permanence; it can do it even better and in less steps if, additionally, active vision is used.

The benefit of the active vision system could be even greater in more complex scenes where the camera can be moved not only on a 1-dimensional trajectory, but in 2 or even 3 dimensions instead: for example, the current approach does not (and does not have to) care about objects that could block the path of the camera. However, this could be the case in reality and we conjecture that approaches that allow more degrees of freedom would also learn ways to prevent collisions with the environment.

One noteworthy finding of the evaluation is the fact that seeing a larger number of potentially redundant frames does not necessarily decrease performance: one could assume that this could prevent the system from focusing on or memorizing the important features. That this does not happen becomes clear when comparing the success rate of the $+1$-policy with that of the $+2$-policy: the performance of the $+1$-policy after 12 frames is about the same as the performance of the $+2$-policy after 6 frames.

V. Conclusions

We have presented a CNN-RNN-based computer vision system inspired by neuroscientific results that is able to achieve object permanence for improved scene understanding, as is evident in its high success rate in answering detailed questions about present objects and their relationships. This is beneficial for example for autonomous robots when grasping objects in unconstrained environments cohabited by humans. We have further shown how this multi-headed system could be improved by active vision through a RL component that utilises the reduction in loss of the system's supervised output heads for its reward signal and learns to act in a way that minimizes remaining uncertainty in the scene embedding with each time step. Evaluation on simple scenes shows that actively controlling the camera significantly outperforms other types of camera control.

As all experiments are solely performed on primitive synthetic data, further research is required to evaluate whether the approach also works in more complex scenarios with 3D camera trajectories and ultimately in the real world, especially in cases with heavy noise on the depth channel of the input frames as produced by commercial sensors. Other real-world challenges include data and label availability [38], [2]. Additional future work lies in modifying the reward function to favor trajectories which require less resources (or risk) from an embodied agent.

ACKNOWLEDGEMENTS

We are grateful for the recognition of this work by the GST through the Dr. Waldemar Jucker award 2020.

REFERENCES

[1] J. Aloimonos, I. Weiss, and A. Bandyopadhyay, "Active vision," *International journal of computer vision*, vol. 1, no. 4, pp. 333–356, 1988.

[2] M. Braschler, T. Stadelmann, and K. Stockinger, *Applied Data Science*. Springer, 2019.

[3] B. Burger, P. M. Maffettone, V. V. Gusev, C. M. Aitchison, Y. Bai, X. Wang, X. Li, B. M. Alston, B. Li, R. Clowes, *et al.*, "A mobile robotic chemist," *Nature*, vol. 583, no. 7815, pp. 237–241, 2020.

[4] R. Cheng, A. Agarwal, and K. Fragkiadaki, "Reinforcement learning of active vision for manipulating objects under occlusions," *arXiv preprint arXiv:1811.08067*, 2018.

[5] R. M. Cichy, A. Khosla, D. Pantazis, A. Torralba, and A. Oliva, "Comparison of deep neural networks to spatio-temporal cortical dynamics of human visual object recognition reveals hierarchical correspondence," *Scientific reports*, vol. 6, 2016.

[6] B. O. Community, *Blender - a 3D modelling and rendering package*, Blender Foundation, Stichting Blender Foundation, Amsterdam, 2020. [Online]. Available: http://www.blender.org

[7] S. Dasari, F. Ebert, S. Tian, S. Nair, B. Bucher, K. Schmeckpeper, S. Singh, S. Levine, and C. Finn, "Robonet: Large-scale multi-robot learning," *arXiv preprint arXiv:1910.11215*, 2019.

[8] G. de Croon, S. Nolfi, and E. O. Postma, "Towards pro-active embodied agents: On the importance of . . ." in *COMPLEX ENGINEERING SYSTEMS. PERSEUS BOOKS GROUPS*. Press, 2004.

[9] G. de Croon, I. Sprinkhuizen-Kuyper, and E. Postma, "Comparing active vision models," *Image and Vision Computing*, vol. 27, no. 4, pp. 374 – 384, 2009. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0262885608001315

[10] J. Drewes, G. Goren, W. Zhu, and J. H. Elder, "Recurrent processing in the formation of shape percepts," *Journal of Neuroscience*, vol. 36, no. 1, pp. 185–192, 2016.

[11] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: part i," *IEEE robotics & automation magazine*, vol. 13, no. 2, pp. 99–110, 2006.

[12] S. A. Eslami, N. Heess, T. Weber, Y. Tassa, D. Szepesvari, G. E. Hinton, *et al.*, "Attend, infer, repeat: Fast scene understanding with generative models," in *Advances in Neural Information Processing Systems*, 2016, pp. 3225–3233.

[13] S. A. Eslami, D. J. Rezende, F. Besse, F. Viola, A. S. Morcos, M. Garnelo, A. Ruderman, A. A. Rusu, I. Danihelka, K. Gregor, *et al.*, "Neural scene representation and rendering," *Science*, vol. 360, no. 6394, pp. 1204–1210, 2018.

[14] R. Ewerth, C. Beringer, T. Kopp, M. Niebergall, T. Stadelmann, and B. Freisleben, "University of marburg at trecvid 2005: Shot boundary detection and camera motion estimation results." in *TRECVID*, 2005.

[15] J. Foerster, N. Nardelli, G. Farquhar, T. Afouras, P. H. S. Torr, P. Kohli, and S. Whiteson, "Stabilising experience replay for deep multi-agent reinforcement learning," in *Proc. International Conference on Machine Learning*, 2017, p. 1146–1155.

[16] M. Gori, "What's wrong with computer vision?" in *IAPR Workshop on Artificial Neural Networks in Pattern Recognition*. Springer, 2018, pp. 3–16.

[17] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick, "Mask R-CNN," *Proc. IEEE International Conference on Computer Vision (ICCV)*, pp. 2980–2988, 2017.

[18] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[19] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[20] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[21] S. Kastner and M. Pinsk, "Visual attention as a multilevel selection process," *Cognitive, affective and behavioral neuroscience*, vol. 4, pp. 483–500, 01 2005.

[22] T. Kato and D. Floreano, "An evolutionary active-vision system," in *Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat. No.01TH8546)*, vol. 1, 2001, pp. 107–114 vol. 1.

[23] V. R. Konda and J. N. Tsitsiklis, "Actor-critic algorithms," in *Advances in Neural Information Processing Systems*, S. A. Solla, T. K. Leen, and K. Müller, Eds., 2000, pp. 1008–1014.

[24] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.

[25] Y. LeCun, C. Cortes, and C. Burges, "Mnist handwritten digit database," *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, vol. 2, 2010.

[26] D. Marocco and D. Floreano, "Active vision and feature selection in evolutionary behavioral systems," *From Animals to Animats 7: Proceedings of the Seventh International Conference on Simulation of Adaptive Behavior*, pp. 247–255, 2002. [Online]. Available: http://infoscience.epfl.ch/record/63940

[27] N. K. Medathati, H. Neumann, G. S. Masson, and P. Kornprobst, "Bio-inspired computer vision: Towards a synergistic approach of artificial and biological vision," *Computer Vision and Image Understanding*, vol. 150, pp. 1–30, 2016.

[28] C. Merkel, J.-M. Hopf, and M. A. Schoenfeld, "How to perceive object permanence in our visual environment: The multiple object tracking paradigm," *Spatial Learning and Attention Guidance*, pp. 157–176, 2020.

[29] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International conference on machine learning*, 2016, pp. 1928–1937.

[30] V. Mnih, N. Heess, A. Graves, *et al.*, "Recurrent models of visual attention," in *Advances in neural information processing systems*, 2014, pp. 2204–2212.

[31] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[32] P. G. Pawar and V. Devendran, "Scene understanding: A survey to see the world at a single glance," in *Proc. IEEE International Conference on Intelligent Communication and Computational Techniques*, 2019, pp. 182–186.

[33] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 779–788.

[34] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," in *Advances in neural information processing systems*, 2017, pp. 3856–3866.

[35] G. E. Schneider, "Two visual systems," *Science*, vol. 163, no. 3870, pp. 895–902, 1969.

[36] M. W. Self and P. R. Roelfsema, "Neuroscience: Figured out by feedback to the thalamus," *Current Biology*, vol. 29, no. 12, pp. R574 – R577, 2019.

[37] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[38] T. Stadelmann, M. Amirian, I. Arabaci, M. Arnold, G. F. Duivesteijn, I. Elezi, M. Geiger, S. Lörwald, B. B. Meier, K. Rombach, and L. Tuggener, "Deep learning in the wild," in *Proc. IAPR Workshop on Artificial Neural Networks in Pattern Recognition*, 2018, pp. 17–38.

[39] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*, 2nd ed. Cambridge, MA: MIT Press, 2018.

[40] L. Tuggener, I. Elezi, J. Schmidhuber, and T. Stadelmann, "Deep watershed detector for music object recognition," in *Proc. International Society for Music Information Retrieval Conference*, 2018.

[41] B. Xu, N. Wang, T. Chen, and M. Li, "Empirical evaluation of rectified activations in convolutional network," *arXiv preprint arXiv:1505.00853*, 2015.

[42] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," in *Proc. International Conference on Machine Learning*, 2015, pp. 2048–2057.